**Full Stack Development with MERN**

**Frontend Development Report**

|  |  |
| --- | --- |
| Date | 10 JUL 2024 |
| Team ID | SWTID1720004076 |
| Project Name | Wise Wallet: Your Budgeting Partner |
| Maximum Marks |  |

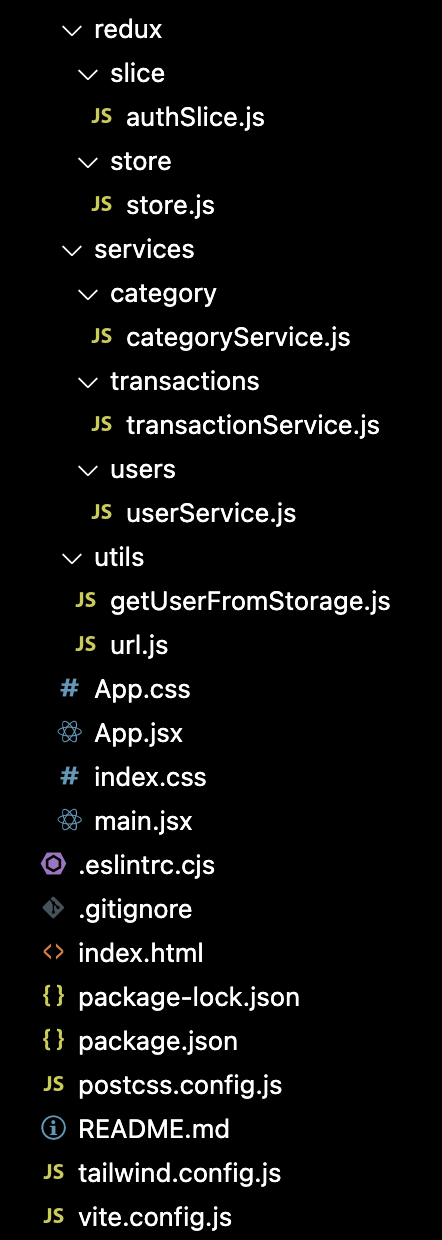
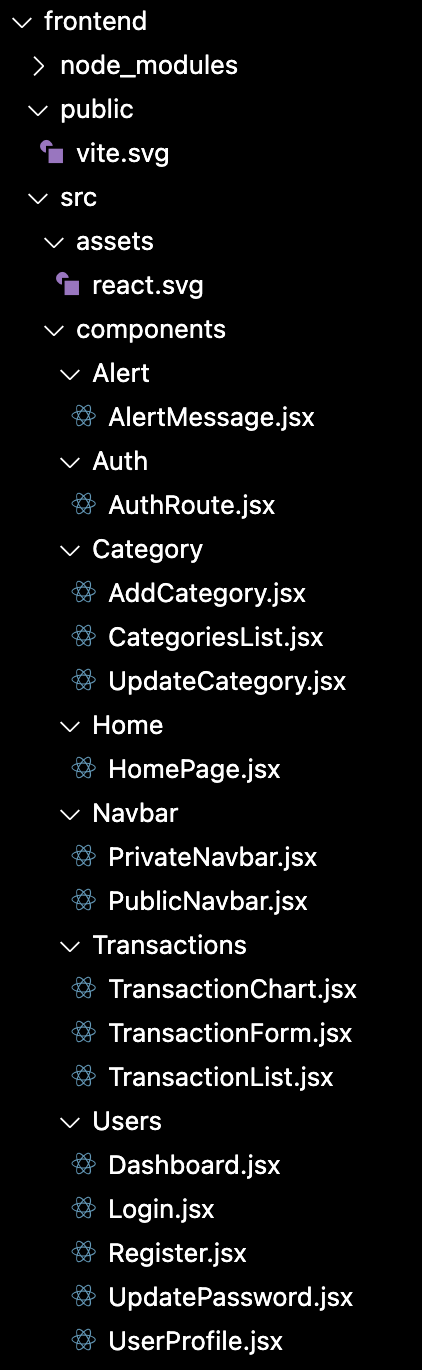
**Objective**

The objective of this report is to document the frontend development progress and key aspects of the user interface implementation for the Wise Wallet: Your Budgeting Partner project.

**Technologies Used**

* **Frontend Framework:** React.js
* **State Management:** Redux,React-Queries
* **UI Framework/Libraries:** ​​Tailwind CSS
* **API Libraries:** Axios

**Project Structure**

Project Directory:

**Explanation:**

**- public:** Contains static files and assets.

**- src:** Main source directory.

**- assets:** Contains asset files like images.

**- components:** Contains reusable UI components.

**- Alert:** Components for displaying alert messages.

**- Auth:** Authentication-related components.

**- Category:** Components for managing categories.

**- Home:** Homepage component.

**- Navbar:** Public and private navigation bars.

**- Transactions:** Components for managing transactions.

**- Users:** User-related components like login, registration, and profile management.

**- redux:** Redux slices and store configuration.

**- slice:** Contains individual slice files for different parts of the state.

**- store:** Configures and combines slices to create the store.

**- services:** Service files for API calls.

**- category:** Category-related API calls.

**- transactions:** Transaction-related API calls.

**- users:** User-related API calls.

**- utils:** Utility functions.

**- getUserFromStorage.js:** Utility function to get user data from local storage.

**- url.js:** Contains URL constants.

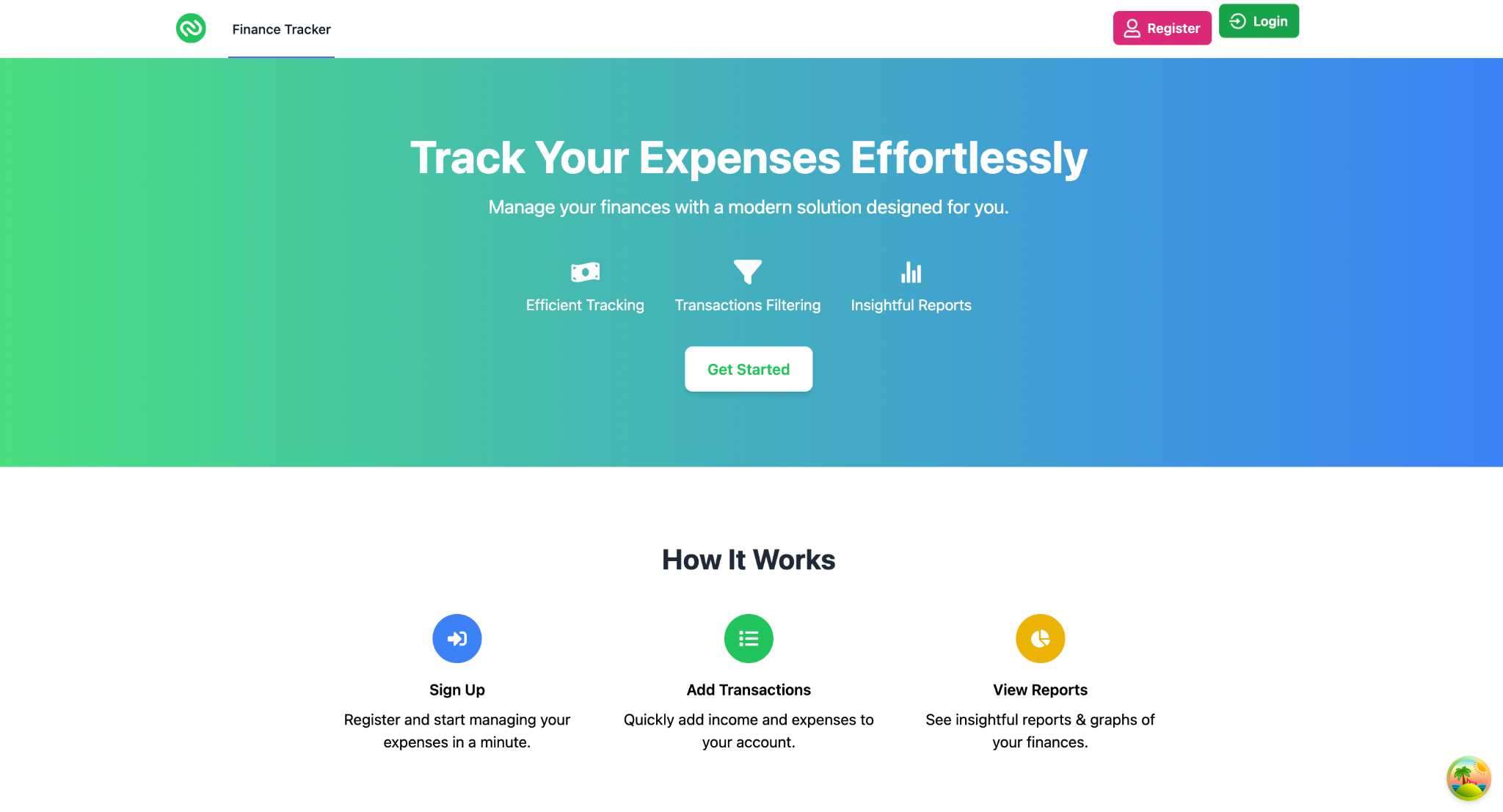
**- App.jsx:** Main application component responsible for routing and layout.

**- index.css:** Global styles.

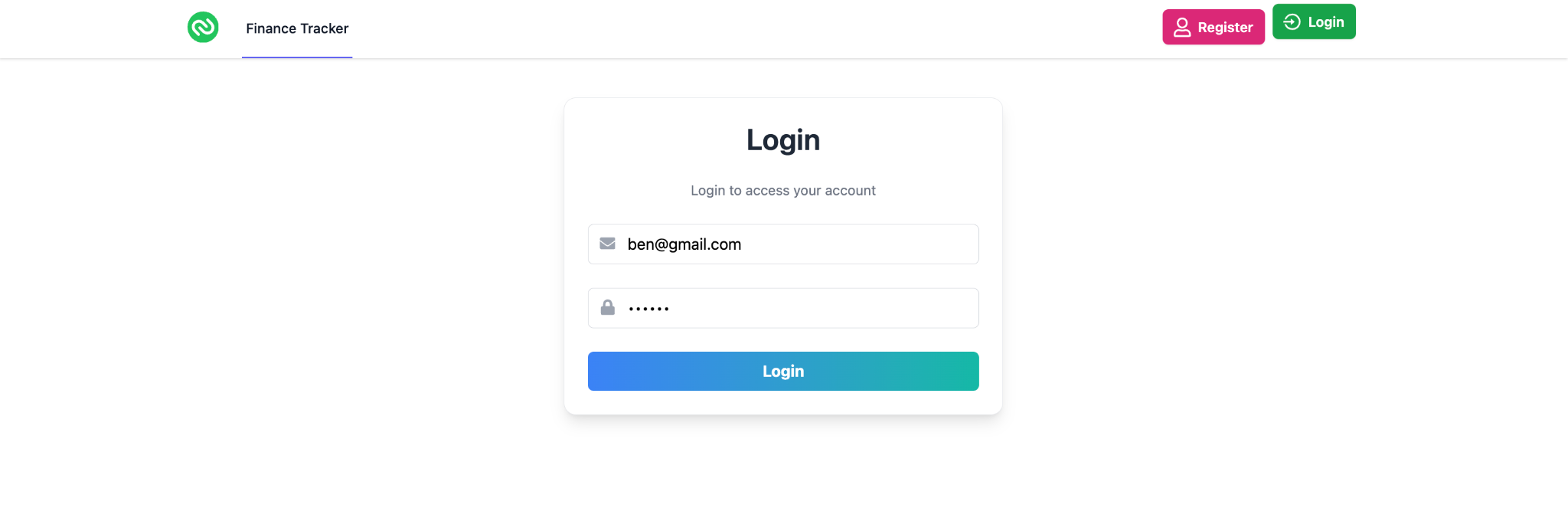
**- main.jsx:** Entry point of the application.

Some screenshots of our running application to show the user interface:

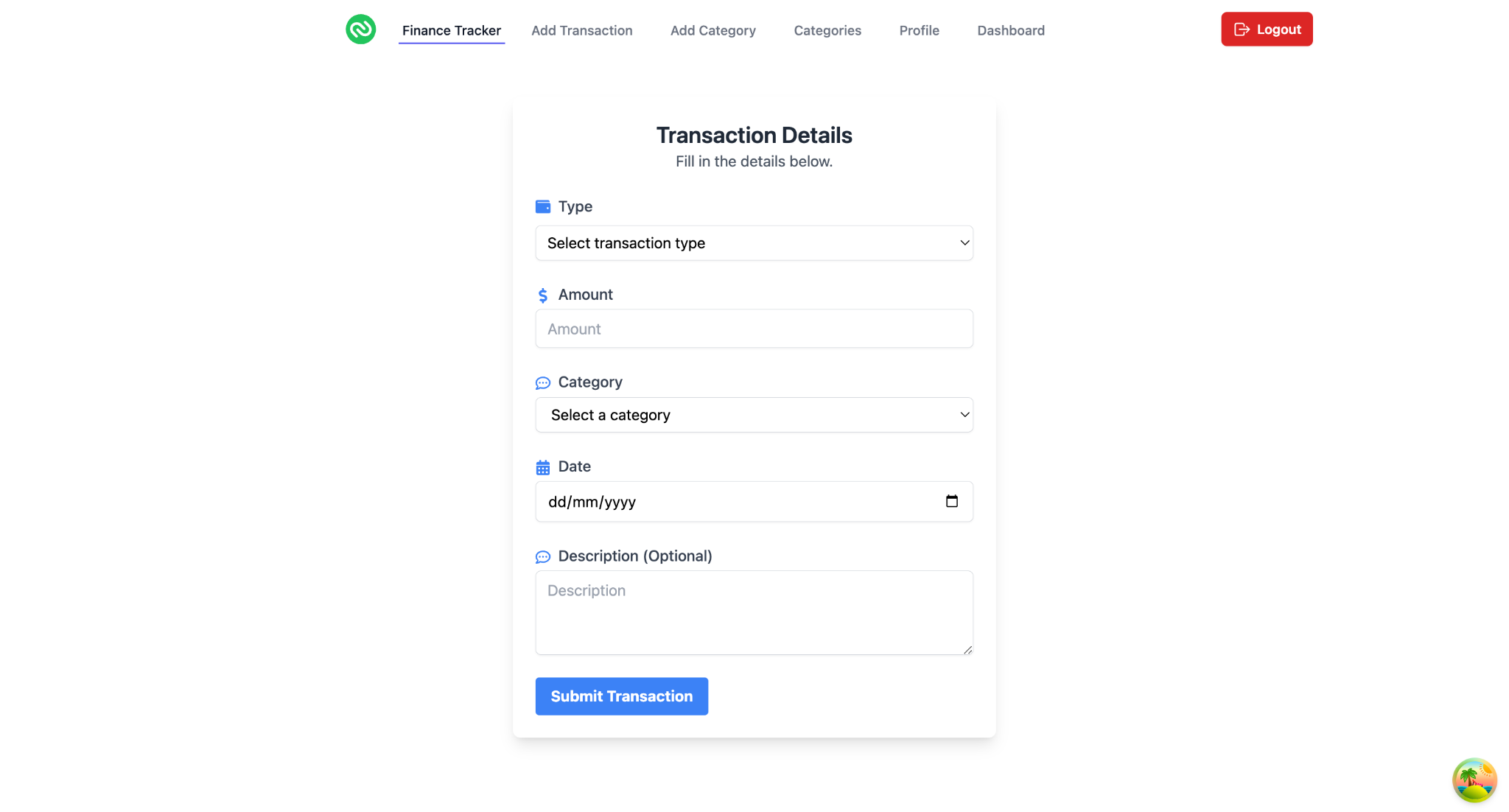
**LANDING PAGE:**

****

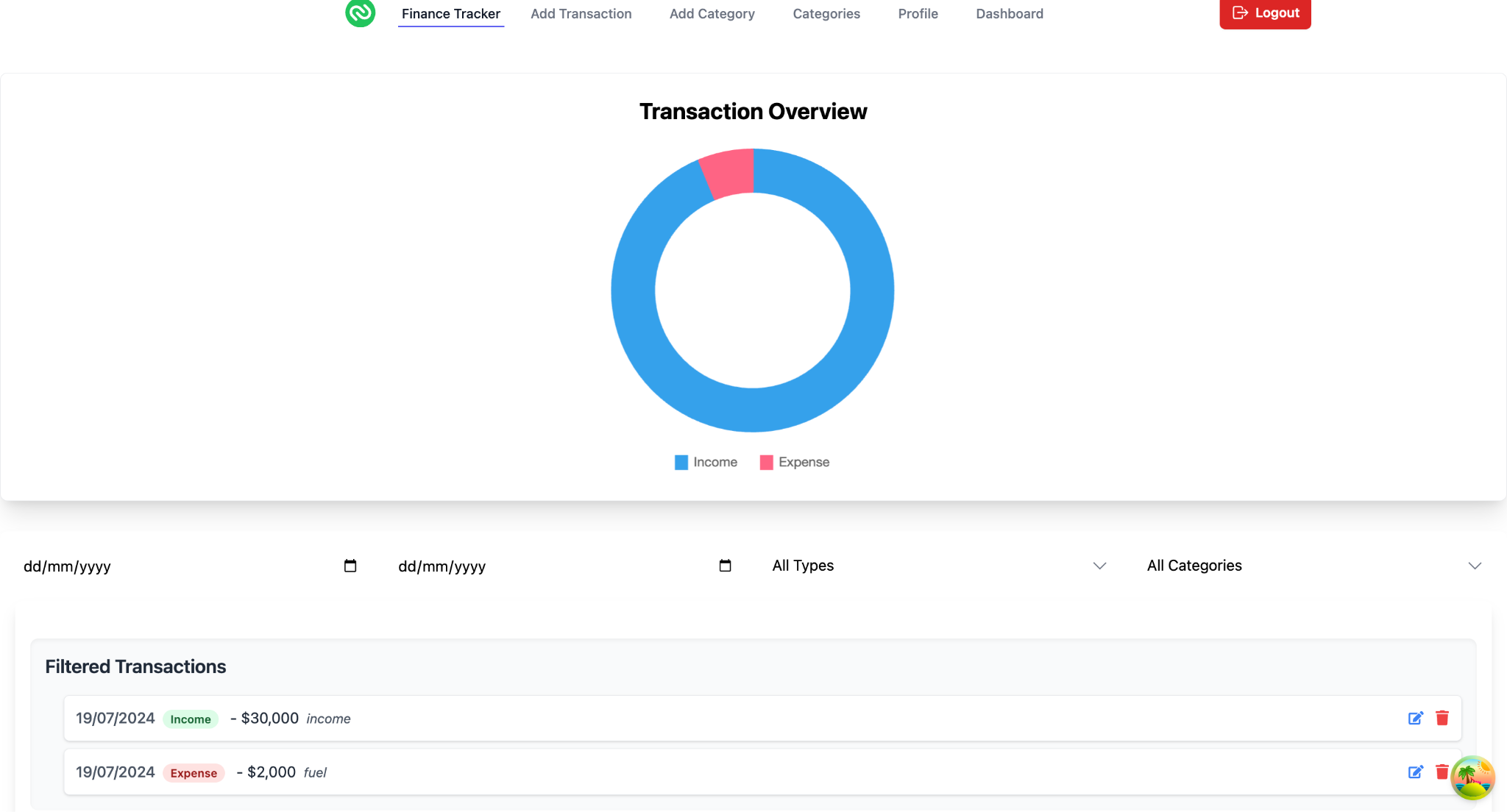
**LOGIN PAGE:**

****

**ADD TRANSACTION PAGE:**

****

**DASHBOARD PAGE:**

****

**Key Components**

1. **App.js -** Responsible for routing and main application layout. Includes private and public routes, based on user authentication.
2. **components**

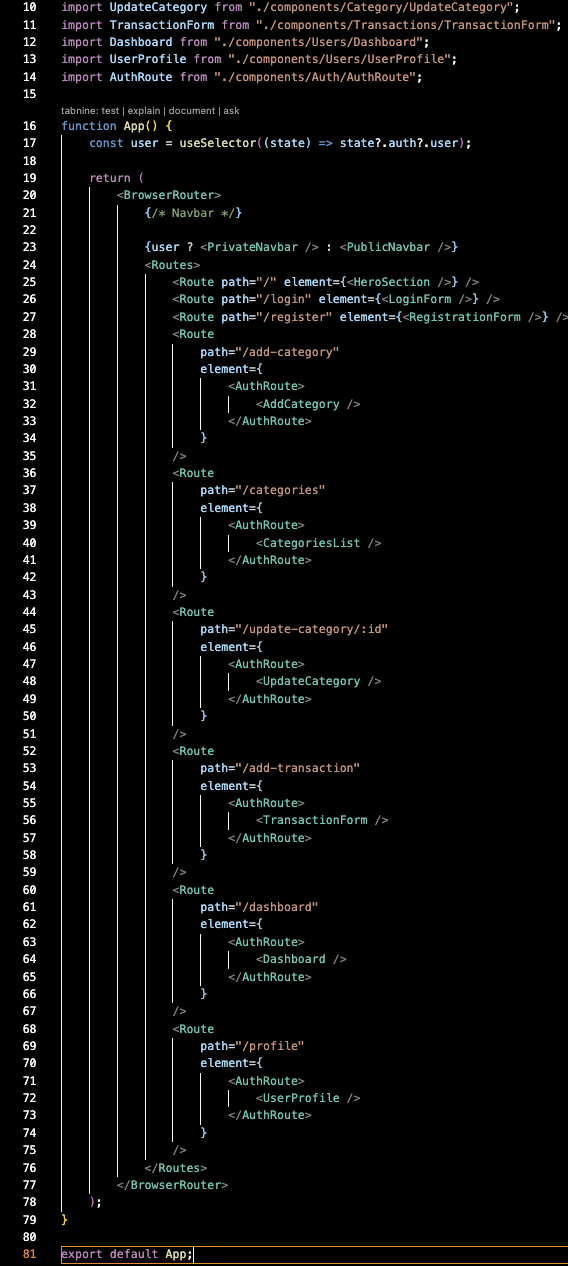
* Contains reusable UI components used across the application, such as:
  + Alert: Displays alert messages.
  + Auth: Contains authentication-related components.
  + Category: Manages category-related components.
  + Home: Contains the homepage component.
  + Navbar: Includes both public and private navigation bars.
  + Transactions: Manages transaction-related components.
  + Users: Manages user-related components like login, registration, profile.

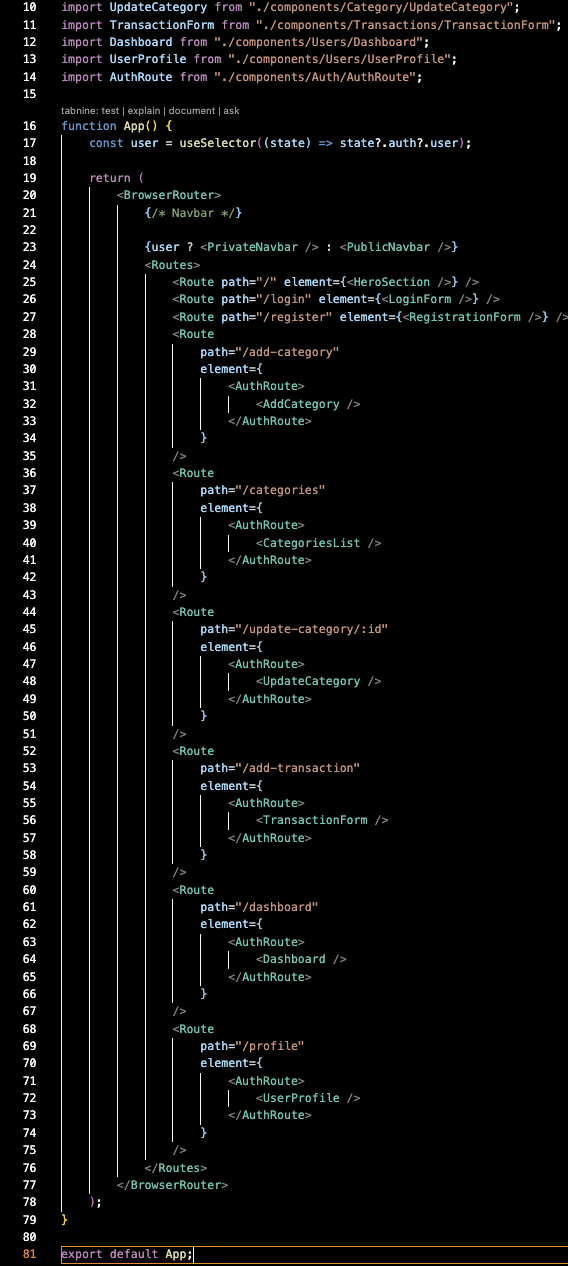
**Routing**

Routing in this project is managed using React Router. Here are the main routes configured in the App.jsx file:

* **/home**: Displays the landing page of the application.
  + **Component**: HomePage.jsx
* **/dashboard**: Shows the user dashboard with data and statistics.
  + **Component**: Dashboard.jsx
* **/profile**: Manages user profile details and settings.
  + **Component**: UserProfile.jsx

ROUTING CONFIGURATION in app.jsx :

****

****

**State Management**

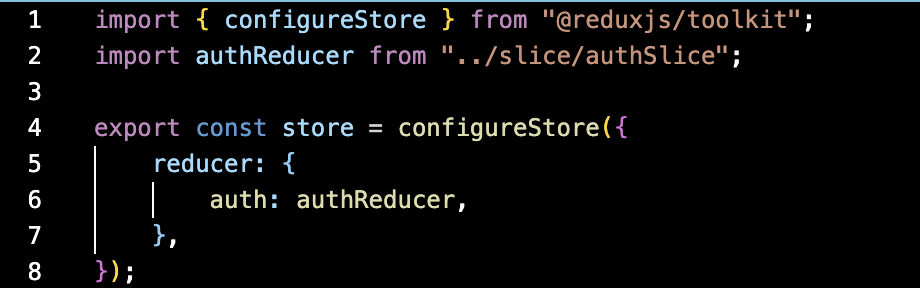
State management is achieved using Redux. The key components include:

* **Redux Store**: Configured in store.js
* **Reducers/Slices**: Manage state slices, such as authentication state in authSlice.js.

**Key Files:**

* **store.js**: Configures and combines Redux slices.
* **authSlice.js**: Contains authentication-related state management.

STORE CONFIGURATION in store.js :



AUTHSLICE CONFIGURATION IN authSlice.js :



**Integration with Backend**

The frontend communicates with backend APIs for data retrieval and user authentication. The APIs are integrated using service files in the services directory.

**Key Endpoints:**

* **GET /api/data**: Retrieves data for display.
* **POST /api/user/login**: Handles user authentication.

**Service Files:**

* **categoryService.js**: Handles category-related API calls.
* **transactionService.js**: Handles transaction-related API calls.
* **userService.js**: Handles user-related API calls.

API CALL in userService.js :





**User Interface (UI) Design**

The UI design follows modern design principles focused on usability and accessibility. It includes:

* **Consistency**: Uniform styles and components across the application.
* **Responsiveness**: Adaptable to different screen sizes and devices.
* **Accessibility**: Ensuring all users, including those with disabilities, can navigate and interact with the application effectively.

**Implemented using**: Tailwind CSS and React components.

**Third-Party Integrations**

Our project integrates several third-party libraries and services:

* **Axios**: For making HTTP requests to the backend API.
* **React Router**: For client-side routing.
* **Redux Toolkit**: For state management.
* **Tailwind CSS**: For utility-first CSS styling.
* **ESLint**: For linting JavaScript code to ensure code quality.
* **Vite**: For fast and optimized build tooling.